Patent/Publication Number: JP11066041A
Application Number: JP1997231316A
Date Filed: 19970827
Title: CALCULATION PROCESSING METHOD FOR SIMULTANEOUS LINEAR EQUATIONS FOR MEMORY DISTRIBUTED PARALLEL COMPUTER AND PARALLEL COMPUTER
Publication Date: 19990309

[INVENTOR]
Name: NAKANISHI MAKOTO
City: Country:

[ASSIGNEE]
Name: FUJITSU LTD
City: Country:

[FOREIGN PRIORITY]
Country: JP
Date Filed: 19970827
Application No.: JP1997231316A

Intl. Class: G06F001712
Intl. Class: G06F001716

[ABSTRACT]
PROBLEM TO BE SOLVED: To solve simultaneous linear equations at high speed by using a comparatively small memory while efficiently performing transfer between processors. SOLUTION: Blocks arranged at processors are cyclically selected and numbered (1) and these numbered blocks are cyclically processed in the order of numbers so that LU decomposition is performed (2). In the case of matrix product calculation in the LU decomposition, data as the calculation object of matrix product are divided and transferred to the respective processors and by simultaneously performing the calculation of the matrix product at each processor to the divided data and parallel transfer, transfer time is shortened. Next, each processor performs formed substitution concerning the LU decomposed result and transfers the result to the adjacent processor (3). Besides, backward substitution is similarly performed as well. When transferring the result of forward/backward substitution to the other processor, the excessive data of one bit are transferred at least in addition to the transfer data and from these one-bit data, the reception of data is confirmed at the respective processors.

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平11－66041

(43)公開日　平成11年(1999)３月９日

(51)Int.Cl.[6]	識別記号	ＦＩ		
Ｇ０６Ｆ　17/12		Ｇ０６Ｆ	15/324	
17/16			15/347	Ｋ

審査請求　未請求　請求項の数４　ＯＬ　（全 11 頁）

(21)出願番号　特願平9－231316

(22)出願日　平成９年(1997)８月27日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　中西　誠
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(74)代理人　弁理士　長澤　俊一郎　（外１名）

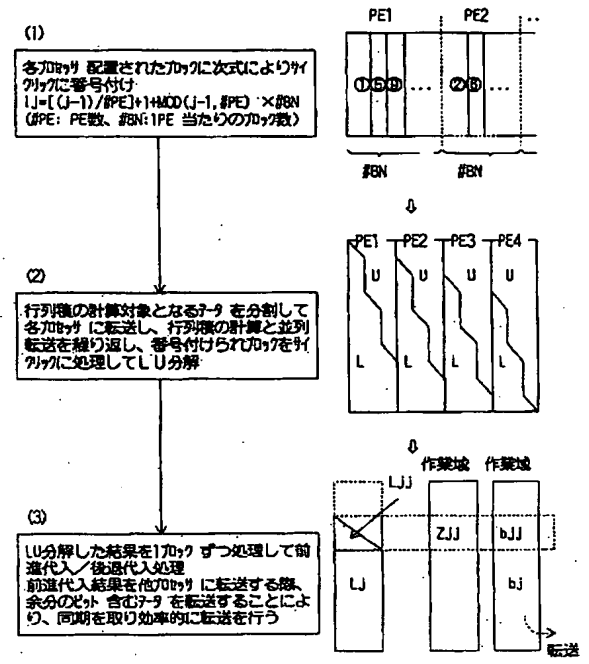
(54)【発明の名称】　メモリ分散型並列計算機による連立一次方程式の計算処理方法および並列計算機

(57)【要約】

【課題】　プロセッサ間で効率のよい転送を行いながら、比較的小さなメモリを使用して高速に連立一次方程式を解くこと。

【解決手段】　プロセッサに配置されたブロックをサイクリックに選択して番号付けし、上記番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解を行う。ＬＵ分解における行列積の計算の際、行列積の計算対象となるデータを分割して各プロセッサに転送し、分割したデータに対する各プロセッサにおける行列積の計算と並列転送を同時に行い転送時間を短縮する。ついで、ＬＵ分解した結果について各プロセッサで前進代入し、結果を隣接するプロセッサに転送する。また、後退代入も同様に行う。前進代入／後退代入結果を他のプロセッサに転送する際、転送データに加えて少なくとも１ビットの余分なデータを転送し、該１ビットデータにより各プロセッサにおいてデータ受信を確認する。

本発明の原理説明図



【特許請求の範囲】
【請求項１】　各プロセッサ間でデータ転送を行うことができる複数のプロセッサを備えたメモリ分散型並列計算機を用い、各プロセッサに係数行列を分配し、ブロック化した外積形式のＬＵ分解法により連立一次方程式を解くメモリ分散型並列計算機による連立一次方程式の計算処理方法であって、
列ベクトルを束ねたブロックが各プロセッサに分散して配置されているとして、プロセッサに配置されたブロックをサイクリックに選択して番号付けし、
上記番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解を行い、
上記ＬＵ分解した結果について各プロセッサで前進代入を行って結果を隣接するプロセッサに転送するとともに、各プロセッサで後退代入を行って結果を隣接するプロセッサに転送する処理を行うことを特徴とするメモリ分散型並列計算機による連立一次方程式の計算処理方法。
【請求項２】　各プロセッサに配置されたブロックについてＬＵ分解を行う途中で行列積の演算を行う際、行列積の計算対象となるデータを分割して各プロセッサに転送し、
分割したデータに対する各プロセッサにおける行列積の計算と、他の分割したデータについての次の行列積の計算に用いる部分の並列転送を同時に行うことにより、転送時間を短縮することを特徴とする請求項１のメモリ分散型並列計算機による連立一次方程式の計算処理方法。
【請求項３】　前進／後退代入の結果を次の前進／後退代入を行うプロセッサに転送するため、各プロセッサに、転送データ長より少なくとも１ビットだけ大きい送信バッファおよび受信バッファを設け、
前進／後退代入の結果を他のプロセッサに転送する際、上記送信バッファに転送データを格納するとともに送信バッファの上記１ビットの領域に所定の値を設定し、全プロセッサで順次上記バッファに格納されたデータの転送を行い、
受信側プロセッサにおいて、上記受信バッファの上記１ビットの領域が所定の値に設定されたことでデータの受信を確認することを特徴とする請求項１または請求項２のメモリ分散型並列計算機による連立一次方程式の計算処理方法。
【請求項４】　各プロセッサ間でデータ転送を行うことができる複数のプロセッサを備え、各プロセッサに係数行列を分配し、ブロック化した外積形式のＬＵ分解法により連立一次方程式を解くメモリ分散型並列計算機であって、
列ベクトルを束ねたブロックが各プロセッサに分散して配置されているとして、プロセッサに配置されたブロックをサイクリックに選択して番号付けをする手段と、
ＬＵ分解における行列積の計算の際、行列積の計算対象となるデータを分割して各プロセッサに転送し、分割したデータに対する各プロセッサにおける行列積の計算と、他の分割したデータについての次の行列積の計算に用いる部分の並列転送を同時に行う処理を繰り返すことにより、上記番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解するＬＵ分解手段と、
上記ＬＵ分解した結果について各プロセッサで前進代入を行って結果を隣接するプロセッサに転送するとともに、各プロセッサで後退代入を行って結果を隣接するプロセッサに転送する処理を行い、前進／後退代入の結果を他のプロセッサに転送する際、転送データ長より少なくとも１ビットだけ大きい送信バッファに転送データを格納するとともに上記１ビットの領域に所定の値を設定し、全プロセッサで順次上記バッファに格納されたデータの転送を行い、受信側プロセッサにおいて、受信バッファの上記１ビットの領域が所定の値に設定されたことでデータの受信を確認する前進代入／後退代入処理手段とを備えたことを特徴とするメモリ分散型並列計算機。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、複数のプロセッサ間で通信を行って処理を進めるマルチプロセッサにより、高速に連立一次方程式を解くことかできるメモリ分散型並列計算機による連立一次方程式の計算処理方法およびメモリ分散型並列計算機に関する。
【０００２】
【従来の技術】並列処理により連立一次方程式を解くアルゴリズムとして、ブロック化した外積型のＬＵ分解法が知られている。上記方法は、外積形式のガウスの消去法をブロック化した方法であり、図１６に示す配列ＡをＬＵ分解する場合、ブロック幅をｄとすると、以下のような処理を行う。
【０００３】ｋ番目の処理で、更新部分$A^{(k)}$を次の計算で更新する。
$$A^{(k)} = A^{(k)} - L2^{(K)} \cdot U2^{(K)} \quad (1)$$
ｋ＋１番目の処理では、$A^{(k)}$を幅ｄで分解してｄだけ小さいマトリックスを同じ式で更新する。$L2^{(K)}$、$U2^{(K)}$は以下の式で求める必要がある。上記（１）式で更新を行うとき、$B^{(K)} = [(L1^{(K)})^T, (L2^{(K)})^T]$ $U1^{(K)}$と分解し、$U'2^{(K)} = (L1^{(K)})^{-1} \cdot U2^{(K)}$と更新する。
【０００４】上記ブロック化した外積型のＬＵ分解法をメモリ分散型並列計算機で実行する場合には、各プロセッサの負荷ができるだけ均等になるように各プロセッサのメモリにデータを効率よく分配し、プロセッサ間で処理対象データの交換を効率よく行う必要がある。また、問題を解くユーザインタフェースを煩雑にせずに提供する必要がある。
【０００５】そこで、本発明者は、先に、データ転送コストを削減し、並列性を高めて高速化を図ることかでき

るメモリ分散型並列計算機による連立一次方程式の計算処理方法を提案した（特開平７－２７１７６０号公報参照）。上記処理方法は、列ベクトルを束ねたブロックレベルで、データをサイクリックな配置に並列転送で並べかえ、ＬＵ分解時に、行列積の計算と、データ転送を並列的に同時に実行する。そして、ＬＵ分解した結果についてデータを元の配置に戻し、さらに行列を行ベクトル方向に分割した配置になるように並び変え、並列に前進／後退代入処理を行うものである。

【０００６】

【発明が解決しようとする課題】上記従来の計算処理方法は次のような問題点を有していた。

（１）列ブロックをサイクリックに動的に並び換えており、そのための作業域として、係数行列を格納する分だけ余分に持つ必要があった。

（２）ＬＵ分解した結果についてデータを元の配置に戻し、前進／後退代入処理を行う際、さらに行列を行ベクトル方向に分割した配置になるように並び換える必要があった。

本発明は上記した事情を考慮してなされたものであって、列ブロックを並び換えることなく、各計算機に分割されて配置されたブロックをサイクリックに処理することにより、比較的小さなメモリを使用して高速に連立一次方程式を解くことができ、また、プロセッサ間で効率のよい転送を行うことができるメモリ分散型並列計算機による連立一次方程式の計算処理方法および計算機を提供することである。

【０００７】

【課題を解決するための手段】図１は本発明の原理構成図である。同図に示すように、本発明はメモリ分散型並列計算機を用い、ブロック化した外積形式のＬＵ分解法により連立一次方程式を次のようにして解く。

（１）列ベクトルを束ねたブロックが各プロセッサに分散して配置されているとして、プロセッサに配置されたブロックをサイクリックに選択して番号付けし、上記番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解を行う。

（２）ＬＵ分解における行列積の計算の際、行列積の計算対象となるデータを分割して各プロセッサに転送し、分割したデータに対する各プロセッサにおける行列積の計算と、他の分割したデータについての次の行列積の計算に用いる部分の並列転送を同時に行う処理を繰り返すことにより、上記番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解する

（３）上記ＬＵ分解した結果について各プロセッサで前進代入し、結果を隣接するプロセッサに転送する。また、後退代入も各プロセッサのブロックを後退代入して、結果を隣接するプロセッサに転送する。前進代入／後退代入結果を他のプロセッサに転送する際、転送データ長より少なくとも１ビットだけ大きい送信バッファに転送データを格納するとともに上記１ビットの領域に所定の値を設定し、全プロセッサで順次上記バッファに格納されたデータの転送を行い、受信側プロセッサにおいて、受信バッファの上記１ビットの領域が所定の値に設定されたことでデータの受信を確認することにより転送の同期をとる。

【０００８】本発明においては、分割された領域に均等に格納された係数行列のＬＵ分解を行うに際し、ピボット列を残りのどのベクトルから選んでもよいことに着目し、上記のようにプロセッサに配置されたブロックをサイクリックに選択して番号付けし、番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解を行っている。このため、従来例のように列ブロックをサイクリックに動的に並べ換えるための作業域を係数行列を格納する分だけ余分に持つ必要がない。その結果、同じメモリを使用して、約$\sqrt{}$（２）倍の問題を解くことができる。

【０００９】また、本発明においては、ＬＵ分解における行列積の計算の際、行列積の計算対象となるデータを分割して各プロセッサに転送しており、分割したデータに対する各プロセッサにおける行列積の計算と、他の分割したデータについての次の行列積の計算に用いる部分の並列転送を同時に行うことができ、見かけの転送時間は、計算と同時に行うことができない最初の転送時間のみとなる。このため、従来の２分木転送に比べ、大幅に転送時間を短縮することができる。すなわち、従来の２分木転送ではＬＯＧ２（＃ＰＥ）に比例した転送時間が掛かるが、本発明においては、プロセッサ数が多くなった場合、計算を行うに必要な転送時間をほぼ１（１つのプロセッサから各プロセッサにデータを転送する時間が、１つのプロセッサから他の１つのプロセッサにデータを転送するに要する時間と同程度の時間となる）とすることができる。

【００１０】さらに、本発明においては、ＬＵ分解された結果が各プロセッサにサイクリックに配置されており、前記した従来例のようにＬＵ分解された結果を行ベクトル方向に並べ換えることなく、ｉ番目のブロックがあるプロセッサで前進代入を行って結果を次の前進代入を行うプロセッサに転送し、後退代入についても、同様に各プロセッサのブロックを後退代入して、結果をプロセッサに転送することにより先進代入／後退代入処理を行う。また、上記データ転送を行う際、前記したように転送データに加えて少なくとも１ビットの余分なデータを転送し、該１ビットデータによりデータ受信を確認しているので、効率的な転送を行うことができる。

【００１１】

【発明の実施の形態】図２は本発明に用いられるメモリ分散型並列計算機の構成の一例を示す図であり、図３は図２における各プロセッサ（以下ＰＥという）の構成の一例を示す図である。図２において、各プロセッサ（Ｐ

Ｅ）１はクロスバーネットワーク６に接続されている。各プロセッサ１は、それぞれスカラ演算を行うスカラユニット２と、ベクトル演算を行うベクトルユニット３と、主記憶装置４と、ＰＥ間通信ユニット５から構成され、ＰＥ間通信ユニット５により上記クロスバーネットワーク６を介して任意の他のプロセッサとの間で通信を行う。

【００１２】各プロセッサ１は図３に示すように構成されている。各スカラユニット２は主記憶装置４のデータを一時的に保持するキャッシュメモリ１１と、演算に用いる汎用レジスタ／浮動少数点レジスタ１２と、スカラ命令を実行するスカラ演算機１３から構成される。また、ベクトルユニット３は、主記憶装置４からデータをロードするためのロードパイプライン１４と、主記憶装置４へデータをストアするためのストアパイプライン１５と、ベクトル演算の対象となる一連のデータを保持するベクトルレジスタ１６と、特定の演算対象をマスクするマスクレジスタ１７と、演算対象データを指定するマスクパイプライン１８と、ベクトルデータの乗算を実行する乗算パイプライン１９と、ベクトルデータの加減算または論理演算を実行する加算／論理演算パイプライン２０と、ベクトルデータの除算を行う除算パイプライン２１から構成される。そして、主記憶装置４がフェッチした命令がベクトル命令であるときに上記ベクトルユニット３が起動され、ベクトル演算が実行される。

【００１３】次に、本発明の実施例の連立一次方程式の解法について説明する。

（１）ＬＵ分解

行列Ａに対する連立一次方程式Ａｘ＝ｂを解くことを考える。ここで、Ａの列ベクトルをｄ本束ねたものをブロックと見なし、この列ベクトルを束ねたブロックを下記のようにＡ１，Ａ２，…，Ａｍとする。

Ａ＝〔Ａ１，Ａ２，…，Ａｍ〕

Ａｊを並べ換えて生成した行列Ｅを下記のように表す。

Ｅ＝〔Ａｉ１，Ａｉ２，…，Ａｉｍ〕

つまり、ＡＰ＝Ｅであり、Ａｘ＝ｂは下記のように表すことかできる。但し、Ｐは列ブロックの並べ換えを表す。

$APP^{-1}x=b$

$EP^{-1}x=b$

【００１４】ここで、$P^{-1}x=y$と置くと、Ａｘ＝ｂは下記の式を解くことに相当する。

Ｅｙ＝ｂ，$P^{-1}x=y$

行列の次数をｎとし、各プロセッサに等しい数のブロックが配置されるとすると、ｎは次のように表される。

ｎ＝＃ＢＮ×＃ＰＥ×ｄ

但し、＃ＰＥは並列プロセッサ数、＃ＢＮは各プロセッサに配置される行列のブロック数、ｄは各ブロックに含まれる列ベクトルの数である。

【００１５】いま、Ｅ＝〔Ａｉ１，Ａｉ２，…，Ａｉｍ〕のi_j（ｊ＝１，…，ｍ）が次の式で与えられる場合について考える。

$$i_1=\{(j-1)/\#PE\}+1+MOD(j-1,\#PE)\times\#BN \cdots (1)$$

ここで、ＭＯＤ（ａ，ｂ）は整数ａを整数ｂで割ったときの剰余を表す。この並び換えに対するP^{-1}は、ブロックをサイクリックに番号付けすることに対応する。すなわち、前記（１）式でi_j番目の行列Ａのブロックに番号ｊを振ることに対応し、各プロセッサのブロック＃ＢＮは図４に示すように番号付けされることになる。

【００１６】ここでは、以上のように番号付けたブロックを並べ換えて作った行列ＥをＬＵ分解することを考えるが、実際には、行列Ａを上記（１）式で決まる番号のブロックに着目してＬＵ分解を行うことになる。行列Ｅのｊ番目のブロックに関する分解は前記したのと同様、図５に示すようになる。すなわち、　行列Ｅのｊ番目のブロックに関する分解は、

$$F^{(j)}=[(L1^{(j)})^T,\ (L2^{(j)})^T]\ U1^{(j)} \cdots (2)$$

と分解し、

$$U'2^{(K)}=(L1^{(j)})^{-1}\cdot U2^{(j)} \cdots (3)$$

と更新する。

【００１７】そして、$E^{(j)}$を次のように更新する。

$$E'^{(j)}=E^{(j)}-L2^{(j)}\cdot U'2^{(j)} \cdots (4)$$

ここで、$L1^{(j)}$，$U1^{(j)}$，$L2^{(j)}$は同一プロセッサ上にあるが、$U2^{(j)}$および$E^{(j)}$は各プロセッサに分散されており、各々のプロセッサ上で計算する必要がある。そこで、$L1^{(j)}$，$L2^{(j)}$を全プロセッサに通信して前記（１）式によりｉｊ＞ｊなるブロックに関して（３）（４）式の計算を行う。

【００１８】図６は上記処理を行うフローチャートである。同図において、ステップＳ１において、ｊ＝１に設定し、ステップＳ２において、行列Ａの前記（１）式で求めたｉｊ番目のブロックを前記（２）式のように分解する。なお、このブロックは、ＭＯＤ（ｊ－１，＃ＰＥ）＋１番目のプロセッサ（ＰＥ）上にある。ステップＳ３において、次のステップＳ４の計算をするため、$L1^{(j)}$を全プロセッサに通信する。ついで、ステップＳ４において、各プロセッサにおいて、前記（３）式により$U'2^{(j)}$を更新する。ステップＳ５において、$L2^{(j)}$を全プロセッサに通信し、ステップＳ６において、ｉｊ＞ｊなるブロックに対して、前記（４）式により、各プロセッサに割り付けられた部分を計算する。ステップＳ７においてｊ＝ｊ＋１とし、ステップＳ８において、ｊ＞＃ＢＮ×＃ＰＥであるかを調べ、ｊ＞＃ＢＮ×＃ＰＥでない場合には、ステップＳ２に戻り上記処理を繰り返す。また、ｊ＞＃ＢＮ×＃ＰＥの場合には処理を終了する。

【００１９】その結果、各プロセッサに配置されたブロ

ックは、図７に示すようにＬＵ分解される。同図において、斜線部分がＬ、斜線の無い部分がＵである。以上のような処理を行うことにより、前記したように、データの並べ換えを行うことなくＬＵ分解を行うことができ、並べ換えの作業域として、係数行列を格納する分だけ余分に持つ必要がなくなり、比較的小さなメモリを使用して連立一次方程式を解くことができる。

【００２０】ところで、上記処理においてはデータを各プロセッサ（ＰＥ）に転送して行列積の計算を行っているが、その際、転送時間を短縮し転送と計算を同時に行うため、前記した特開平７－２７１７６０号で示したように、各プロセッサ（ＰＥ）にデータを分割して転送する。そして、分割したデータに対する計算を各プロセッサ（ＰＥ）で行い、これを繰り返して全体の計算を行う。これにより、実際の転送時間が非常に少なくなったように見える。

【００２１】以下、簡単に上記転送・計算処理について説明する。なお、詳細は上記特開平７－２７１７６０号公報を参照されたい。前記したように、ブロック化したＬＵ分解のｊ番目のステージにおいて、前記した（４）式に示した行列積の演算を行いＥ(J)を更新している。ここでは、プロセッサ数を５として、Ｕ＝Ｕ－Ｃ×Ｒの計算を行ってＵを更新する場合を考える。図８に示すように、行列Ｃを行方向に分割し、例えばＣ１，Ｃ２，Ｃ３とする。なお、上記分割数ｎは、＃ＰＥ／ｎ＞１でＬＯＧ２（＃ＰＥ／ｎ）＜ｎになるように定める。ここでは分割数を３とする。また、各プロセッサに第１のワーク領域Ｗ１１～Ｗ５１、第２のワーク領域Ｗ１２～Ｗ５２を設ける。

【００２２】図８において、ＰＥ１におけるＣの部分の計算が完了したら、Ｃ１をＰＥ１のＷ１１へ、Ｃ２をＰＥ２のＷ２１へ、Ｃ３をＰＥ３のＷ３１へそれぞれ転送する。次に、その結果を使ってＷ１１のデータ（Ｃ１）をＰＥ４のＷ４１へ、Ｗ２１のデータ（Ｃ２）をＰＥ５のＷ５１へ並列転送する。行列式の計算は、最初に各ＰＥの第１のワーク領域Ｗ１１～Ｗ５１に格納されたＣｉを用いて計算を行う。図８においては、ハッチングで示した部分が最初に計算する部分である。すなわち、ＰＥ１においてＣ１×Ｒ１の行列積が計算され、ＰＥ２においてＣ２×Ｒ２の行列積が計算され、ＰＥ３においてＣ３×Ｒ３の行列積が計算され、ＰＥ４においてＣ１×Ｒ４の行列積が計算され、さらに、ＰＥ５においてＣ２×Ｒ５の行列積が計算される。

【００２３】上記計算と同時に、図８に示すようにワーク領域Ｗ１１からＷ２２へ、Ｗ２１からＷ３２へ、Ｗ３１からＷ４２へ、Ｗ４１からＷ５２へ並列にデータ転送を行うとともに、ＰＥ１が保持するＣ３をＷ１２に転送する。次に、各ＰＥの第２のワーク領域Ｗ１２～Ｗ５２に格納されたＣｉを使って、図９に示すハッチングの部分を計算する。すなわち、ＰＥ１においてＣ３×Ｒ１の行列積が計算され、ＰＥ２においてＣ１×Ｒ２の行列積が計算され、ＰＥ３においてＣ２×Ｒ３の行列積が計算され、ＰＥ４においてＣ３×Ｒ４の行列積が計算され、さらに、ＰＥ５においてＣ１×Ｒ５の行列積が計算される。

【００２４】上記計算と同時に、図９に示すようにワーク領域Ｗ１２からＷ２１へ、Ｗ２２からＷ３１へ、Ｗ３２からＷ４１へ、Ｗ４２からＷ５１へ並列にデータ転送を行うとともに、ＰＥ１が保持するＣ２をＷ１１に転送する。さらに、各ＰＥの第１のワーク領域Ｗ１１～Ｗ５１に格納されたＣｉを使って、図１０に示すハッチングの部分を計算する。すなわち、ＰＥ１においてＣ２×Ｒ１の行列積が計算され、ＰＥ２においてＣ３×Ｒ２の行列積が計算され、ＰＥ３においてＣ１×Ｒ３の行列積が計算され、ＰＥ４においてＣ２×Ｒ４の行列積が計算され、さらに、ＰＥ５においてＣ３×Ｒ５の行列積が計算される。

【００２５】以上のようにしてＰＥ１に保持されたＣ１～Ｃ３についての行列積の計算を終了すると、ついで、図１１のハッチングで示す部分について、ＬＵ分解して、上記と同様、行列積の計算を行う。すなわち、図１２に示すようにＰＥ２に保持されたＣ１～Ｃ３を３分割し、Ｃ１をＰＥ２のワーク領域Ｗ２１へ、Ｃ２をＰＥ３のＷ３１へ、Ｃ３をＰＥ４のＷ４１へそれぞれ転送し、その結果を使って、Ｗ２１（Ｃ１）のデータをＰＥ５のＷ５１へ、Ｗ３１（Ｃ２）のデータをＰＥ１のＷ１１に転送する。

【００２６】次に、前記したようにして各ＰＥの第１のワーク領域Ｗ１１～Ｗ５１に格納されたＣｉを使って、図１２のハッチングで示した部分の計算を行い、これらの計算と同時に、図１２に示すように、ワーク領域Ｗ２１からＷ３２へ、Ｗ３１からＷ４２へ、Ｗ４１からＷ５２へ、Ｗ５１からＷ１２へ並列にデータ転送を行うとともに、ＰＥ２が保持するＣ３をワーク領域Ｗ２２に転送する。以下、同様に計算と転送を繰り返し、ＬＵ分解を完了する。

【００２７】以上のように、行列積部分の計算に必要なデータを分割して、転送・計算することにより、転送の大部分を計算と同時に行うことができる。この場合、計算と転送を同時に行うことができない最初の転送のみが見掛け上の転送時間となるが、この転送も並列に行うようにしたため、従来の２分木転送に比べ大幅に転送時間を短縮することができる。その結果、単純に行列積の計算で必要なデータを各プロセッサに２分木のパターンで転送する場合の転送時間がＬＯＧ２（＃ＰＥ）に比例するのに比べ、本実施例では、１＋〔ＬＯＧ２（＃ＰＥ／＃ｄｉｖ）〕／＃ｄｉｖ（＃ｄｉｖは分割数）のオーダになる。上記式の第２項は０．５以下にすることができるので、プロセッサ数が２台以上あるシステムで、特にプロセッサ数が大きくなった場合に効率を著しく向上さ

せることができる。

【００２８】（２）前進代入／後退代入処理

上記ＬＵ分解の処理結果は、図６に示したように行列Ａのブロックに前記（１）式で指定される順番に格納されている。本実施例の前進代入／後退代入処理では、前記した特開平７－２７１７６０号公報のように行列を行ベクトル方向に並び換えずに、１ブロックずつ処理をする。すなわち、ＬＵ分解したＬＵｙ＝ｂを、Ｌｚ＝ｂ、Ｕｙ＝ｚとし、この順に解く。Ｌｚ＝ｂを解くとき、ｊが１から＃ＢＮ×＃ＰＥまで順に動き、次のような計算を行う。各プロセッサに、図１３に示すように上記ｚおよびｂを格納するためのワーク領域Ｗｚ，Ｗｂを設け、このワーク領域を利用して計算を行う。

【００２９】まず、Ｌjj・Ｚjj＝ｂjjを解き、次にｂｊ＝ｂｊ －Ｌｊ ×Ｚjj によりｂｊ を更新する。すなわち、ＰＥ１でＬ11・Ｚ11＝ｂ11を解き、ｂ1 ＝ｂ1 －Ｌ1 ×Ｚ11 によりｂ1 を更新して、次いで、ＰＥ２において、ｊ＝２について同様の処理を行い、以下同様に各プロセッサにおいて上記処理を繰り返す。上記ｂｊ はｊ＋１番目の処理で必要となるため、他のプロセッサに転送する必要がある。

【００３０】この転送は、ＰＥ１からＰＥ２へ，ＰＥ２からＰＥ３へ，…，ＰＥ＃ＰＥ（＃ＰＥはプロセッサ数）からＰＥ１へと循環するパターンで行われる。上記転送を行うため、各プロセッサに、上記ｂと同じ大きさのバッファｂＴを設け、ｊ番目の前進代入において、バッファｂＴの１＋（ｊ－１）／＃ＰＥ×ｄ番目のデータから最後のデータまで（ｂ（１＋（ｊ－１）／＃ＰＥ×ｄ：ｎ）を循環的に同時に転送する。その際、以下のようにして転送の同期を取る。

【００３１】転送するデータ長は全プロセッサで一定であり、その長さをＬとすると、図１４に示すように、各プロセッサにＬ＋１の受信バッファ、送信バッファを設け、全プロセッサでＬ＋１のデータを転送する。その際、送信バッファのＬ＋１番目には１を設定しておき、受信バッファのＬ＋１番目には０を設定しておく。そして、転送時、全プロセッサで次の図１５に示す処理を行い、データを受け取ったかを確かめる。図１５において、ステップＳ１においてバリア同期を取り、ステップＳ２において、転送データ長Ｌを決めて、図１４（ａ）に示すように受信バッファのＬ＋１番目を０クリアし、送信バッファのＬ＋１番目に１を設定する。ステップＳ３において、全プロセッサでＬ＋１のデータを送り、ステップＳ４において、受信バッファのＬ＋１番目が１であるかを調べ、図１４（ｂ）に示すように受信バッファのＬ＋１番目が１になるとデータを受け取ったとして、次の処理を行う。

【００３２】上記のようにして前進代入を行った後、後退代入においても、各プロセッサにブロックを後退代入して、結果を隣接するプロセッサに転送することで解く。以上のように、本実施例においては前進代入／後退代入処理において、長さＬのデータの転送する際、全プロセッサでＬ＋１番目の余分のデータを転送し、受信したプロセッサにおいて、Ｌ＋１番目のデータを参照してデータ受信を確認しているので、効率のよい転送を行うことができる。

【００３３】

【発明の効果】以上説明したように、本発明においては以下の効果を得ることができる。

（１）プロセッサに配置されたブロックをサイクリックに選択して番号付けし、番号付けられたブロックを、番号順にサイクリックに処理をしてＬＵ分解を行うことによりプロセッサの負荷を均等化している。このため、列ブロックをサイクリックに動的に並べ換えるための作業域を係数行列を格納する分だけ余分に持つ必要がなく、同じメモリを使用して、約√（２）倍の問題を解くことができる。

【００３４】（２）ＬＵ分解における行列積の計算の際、行列積の計算対象となるデータを分割して、転送・計算を行っており、転送の大部分を計算と同時に行うことができ、見かけの転送時間は、計算と同時に行うことができない最初の転送時間のみとなる。このため、従来の２分木転送に比べ、大幅に転送時間を短縮することができ、計算を行うに必要な転送時間をほぼ１（１つのプロセッサから各プロセッサにデータを転送する時間が、１つのプロセッサから他の１つのプロセッサにデータを転送するに要する時間と同程度の時間となる）とすることができる。特にプロセッサ数が多くなった場合に非常に効率よい処理を行うことができる。

（３）先進代入／後退代入処理において、データを他のプロセッサに転送する際、転送データに加えて少なくとも１ビットの余分なデータを転送し、該１ビットデータによりデータ受信を確認しているので、効率的な転送を行うことができる。

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】本発明の実施例のメモリ分散型並列計算機の構成例を示す図である。

【図３】図２における各プロセッサの構成を示す図である。

【図４】各プロセッサに配置されたブロックへの番号付けを説明する図である。

【図５】本発明の実施例におけるＬＵ分解を説明する図である。

【図６】本発明の実施例におけるＬＵ分解処理のフローチャートである。

【図７】本発明の実施例による行列ＡのＬＵ分解結果を示す図である。

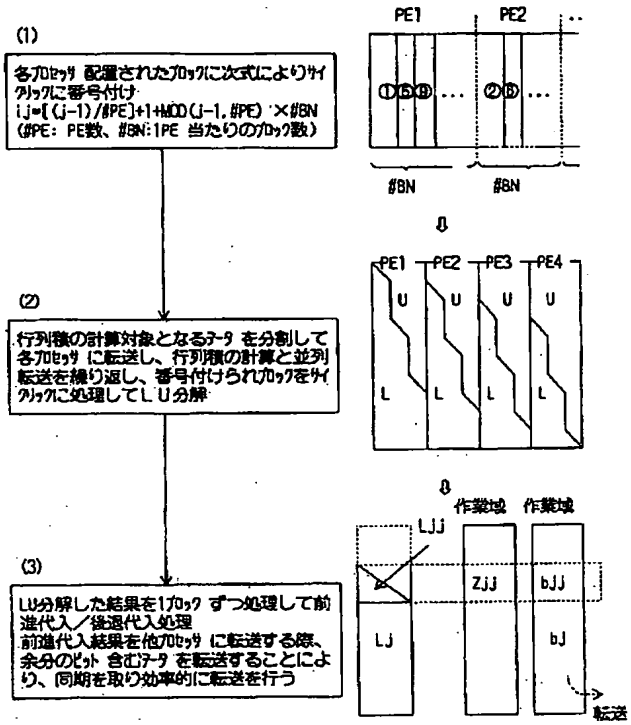
【図８】本発明の実施例における行列積の計算を説明する図（１）である。

【図９】本発明の実施例における行列積の計算を説明する図（２）である。
【図１０】本発明の実施例における行列積の計算を説明する図（３）である。
【図１１】本発明の実施例における行列積の計算を説明する図（４）である。
【図１２】本発明の実施例における行列積の計算を説明する図（５）である。
【図１３】本発明の実施例における前進代入処理を説明する図である。
【図１４】前進代入／後退代入処理時のデータ転送を説明する図である。
【図１５】前進代入／後退代入処理時のデータ転送処理を示すフローチャートである。
【図１６】ブロック化した外積型のＬＵ分解法の説明図である。
【符号の説明】

１　プロセッサ（ＰＥ）
２　スカラユニット
３　ベクトルユニット
４　主記憶装置
５　ＰＥ間通信ユニット
６　クロスバーネットワーク
１１　キャッシュメモリ
１２　汎用レジスタ／浮動少数点レジスタ
１３　スカラ演算機
１４　ロードパイプライン
１５　ストアパイプライン
１６　ベクトルレジスタ
１７　マスクレジスタ
１８　マスクパイプライン
１９　乗算パイプライン
２０　加算／論理演算パイプライン
２１　除算パイプライン

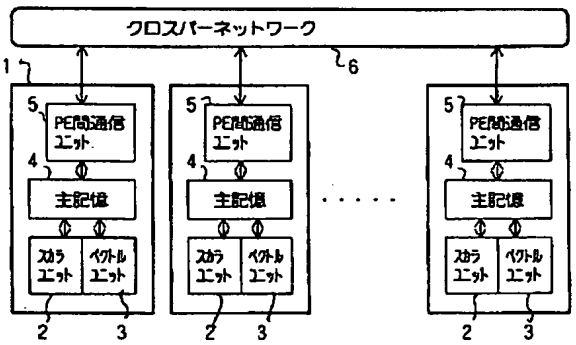
【図１】

本発明の原理説明図



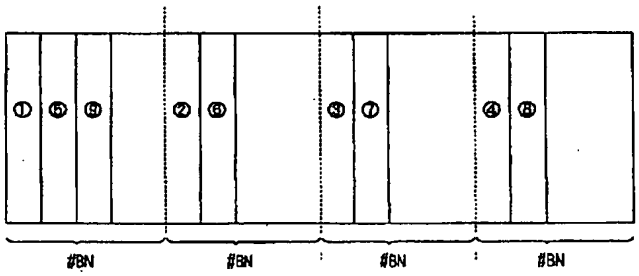
【図２】

本発明の実施例のメモリ分散型並列計算機の構成例を示す図



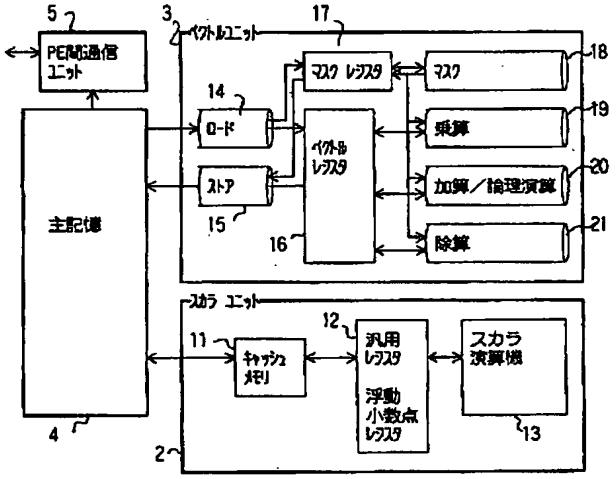
【図４】

各プロセッサに配置されたブロックへの番号付けを説明する図



【図３】

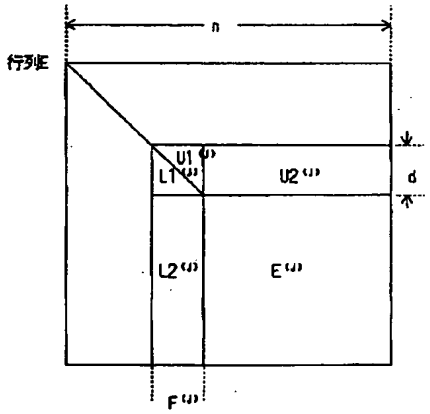
図２における各プロセッサの構成を示す図



【図５】

本発明の実施例におけるＬＵ分解を説明する図

$$l_j = \{(j-1)/\#PE\} + 1 + MOD(j-1, \#PE) \times \#BN \quad \cdots(1)$$

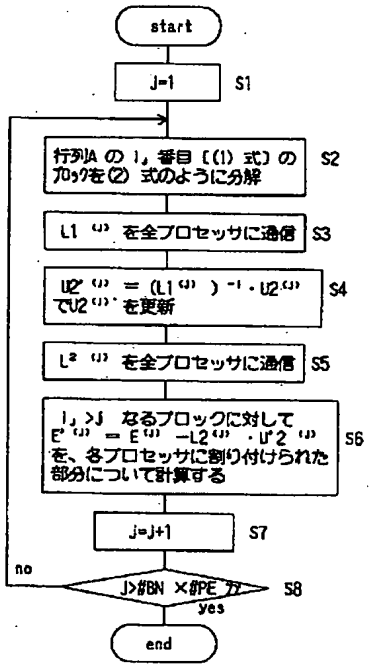


$F^{(j)} = (L1^{(j)T}, L2^{(j)T})^T U1^{(j)}$ … (2)
と分解して

$U2'^{(j)} = (L1^{(j)})^{-1} \cdot U2^{(j)}$ … (3)
と更新

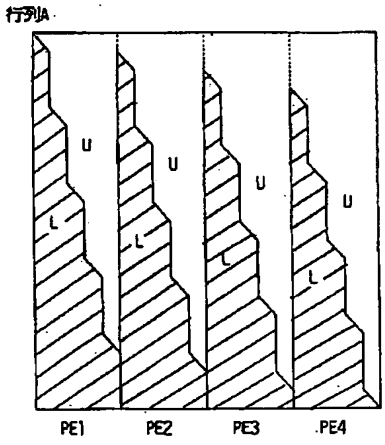
【図６】

本発明の実施例におけるＬＵ分解処理のフローチャート



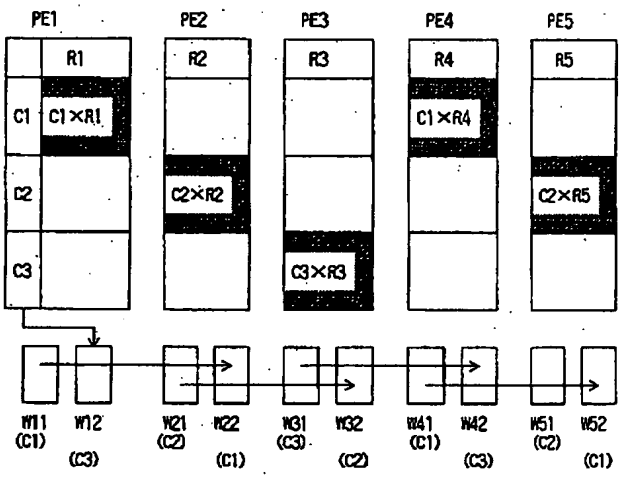
【図７】

本発明の実施例による行列ＡのＬＵ分解結果を示す図



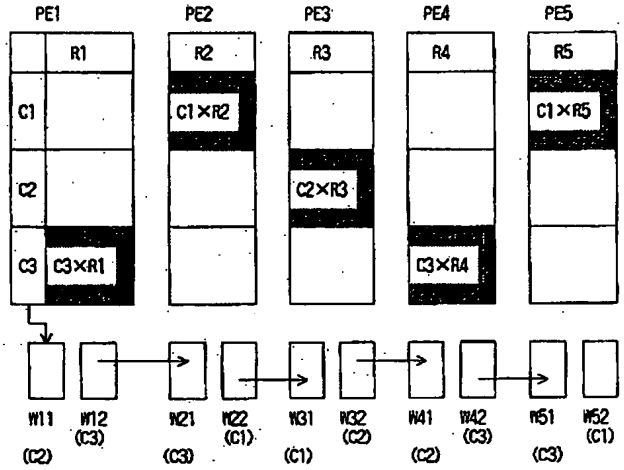
【図８】

本発明の実施例における行列積の計算を説明する図（１）



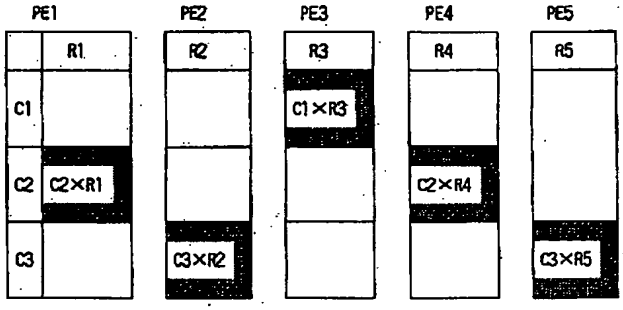
【図９】

本発明の実施例における行列積の計算を説明する図（２）



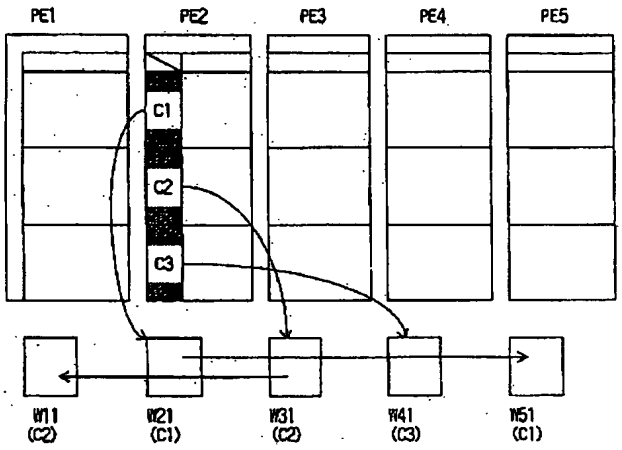
【図１０】

本発明の実施例における行列積の計算を説明する図（３）



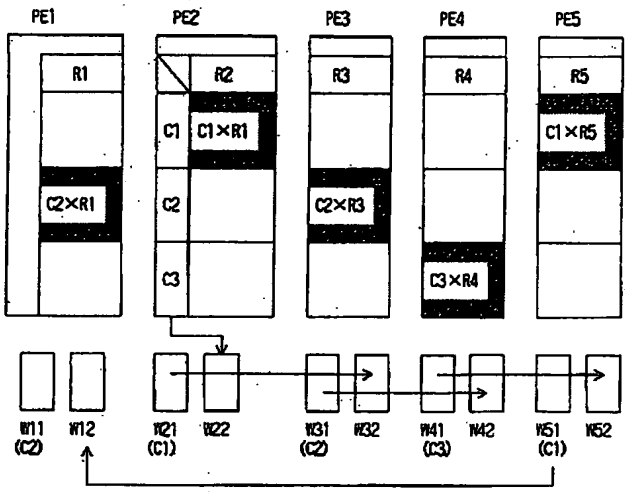
【図１１】

本発明の実施例における行列積の計算を説明する図（４）



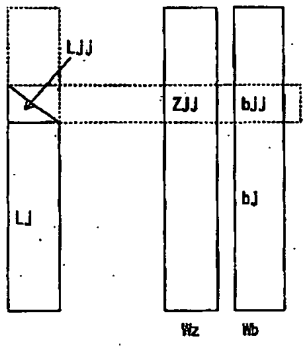
【図１２】

本発明の実施例における行列積の計算を説明する図（５）



【図１３】

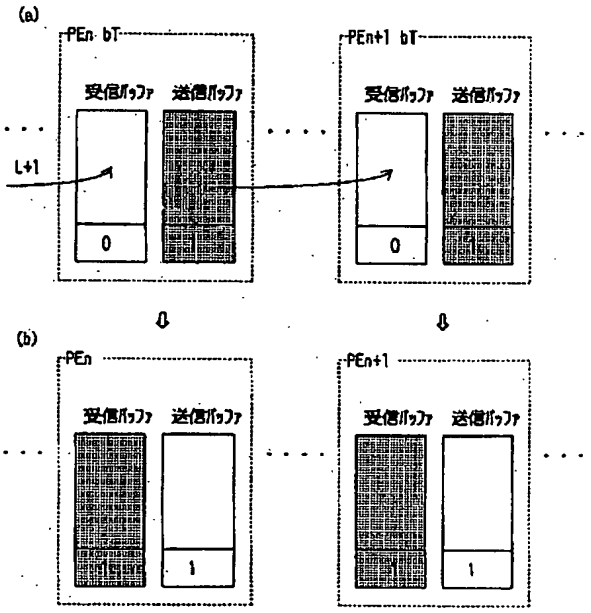
本発明の実施例における前進代入処理を説明する図



LJJ ZJJ =bJJ　を解く
bJ=bJ−LJ×ZJJ でbJ　を更新する

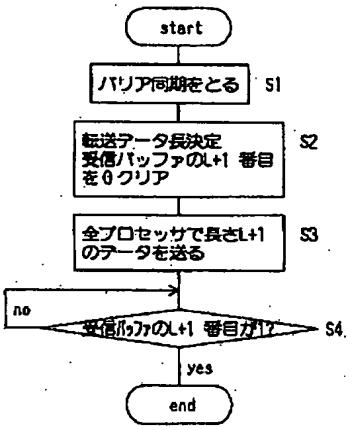
【図１４】

前進代入／後退代入処理時のデータ転送を説明する図



【図１５】

前進代入／後退代入処理時のデータ転送処理を示すフローチャート



【図１６】

ブロック化した外積型のＬＵ分解法の説明図